BaroCRYPT(PostgreSQL) 가이드

목차

1. PostgreSQL 암복호화 모듈

1.1 pgcrypto 모듈

pgcrypto 모듈은 PostgreSQL을 위한 암호화 기능을 제공한다.

이 모듈은 신뢰할 수 있는 모듈로 간주된다. 즉, 현재 데이터베이스에 대한 CREATE 권한이 있는 슈퍼 유저가 아닌 사용자가 설치할 수 있다.

pgcrypto 모듈의 Raw 암호화 함수는 다음과 같다.
- convert_to/convert_from: 문자열 변환/복원
- encode/decode: 16진수 인코딩/디코딩
- encrypt/decrypt: 암호화/복호화

암호화는 utf8로 변환한 후, 암호화 키로 '**aes**' 알고리즘을 사용하여 암호화한 후, 그 값을 16진수(hex)로 encoding 한다.

예) encode(encrypt(convert_to('이름005', 'utf8'), 'ENC_KEY', 'aes'), 'hex')

ENC_KEY는 실제 사용할 암호화 KEY로 대체해야 하며, utf8, aes, hex는 고정값임

복호화는 암호화의 역순. 16진수값을 decoding한 후 암호화를 해제하고, 그 값을 다시 utf8에서 변환한다.

예) convert_from(decrypt(decode(mem_name,'hex'),'ENC_KEY','aes'),'utf8')

ENC_KEY는 실제 사용할 복호화 KEY로 대체해야 하며, utf8, aes, hex는 고정값임

1.2 사전 확인 사항

1) 현재 설치 가능한 모듈 확인

```
[root@baropam ~]# su - postgres
Verification code:

[postgres@baropam ~]$ psql barocryptdb
psql (10.17)
Type "help" for help.


postgres=# select * from pg_catalog.pg_available_extensions;
```

없을 경우 아래와 같이 진행.

2) 설정 파일 찾기

[모듈명].control은 해당 모듈 설치를 위한 설정 파일.

```
$ find / -name pgcrypto.control -print
```

찾을 수 없으면 postgresql-contrib를 다음과 같은 명령어를 통해서 설치해야 한다.

```
$ yum install postgresql-contrib -y
```

3) 확장 모듈 컴파일 및 설치

```
$ cd /root/postgresql-1x.x/contrib/pgcrypto
$ ./configure
$ make -f makefile
$ make install
```

위의 과정을 거치면 다음과 같은 파일이 복사됨.

```
/usr/local/pgsql/share/contrib/pgcrypto.sql
/usr/local/pgsql/share/contrib/uninstall_pgcrypto.sql
/usr/local/pgsql/lib/pgcrypto.so
```

4) 확장 모듈 설치

[중요]해당 DB 관리자로 접속하여 설치.
WI 명령어를 통해 해당 DB의 소유자가 누구인지 확인.
기본적으로 옵션값을 지정해 주지 않는다면 public 스키마 함수들이 만들어짐.

```
postgres=# create extension pgcrypto;
```

5) 현재 설치된 모듈 확인

```
postgres=# select * from pg_catalog.pg_extension;
```

6) 함수 테스트

```
postgres=# select encode(encrypt(convert_to('barocrypt', 'utf-8'), 'utf-8', 'AES'), 'hex');
```

1.3 암복호화 함수

1) TO_ENCRYPTS 함수

- NAME
  TO_ENCRYPTS

- SYNOPSIS
  char TO_ENCRYPTS(data char)
  varchar TO_ENCRYPTS(data varchar)
  txt TO_ENCRYPTS(data text)

- DESCRIPTION
  데이터를 암호화하는 함수
  data: 암호화할 데이터

- RETURN VALUES
  암호화한 데이터를 반환

2) TO_DECRYPTS 함수

- NAME
  TO_DECRYPTS

- SYNOPSIS
  char TO_DECRYPTS(data char)
  varchar TO_DECRYPTS(data varchar)
  txt TO_DECRYPTS(data text)

- DESCRIPTION
  데이터를 복호화 하는 함수
  data: 복호화할 데이터

- RETURN VALUES
  복호화한 데이터를 반환

1.4 암복호화 함수 생성

1) 확장 모듈 설치

```
[root@baropam ~]# su - postgres
Verification code:

[postgres@baropam ~]$ psql barocryptdb
psql (10.17)
Type "help" for help.

postgres=# CREATE EXTENSION pgcrypto;
CREATE EXTENSION
postgres=#
```

2) 암복호화 함수 생성

```
postgres=# CREATE OR REPLACE FUNCTION TO_ENCRYPTS(data char)
postgres-# RETURNS char AS
postgres-# $$ BEGIN
postgres$#     RETURN encode(encrypt(convert_to(data, 'utf8'), '암호화 KEY', 'aes'), 'hex');
postgres$# END; $$
postgres-# LANGUAGE PLPGSQL;
CREATE FUNCTION
postgres=# CREATE OR REPLACE FUNCTION TO_DECRYPTS(data char)
postgres-# RETURNS char AS
postgres-# $$ BEGIN
postgres$#     RETURN convert_from(decrypt(decode(data, 'hex'), '복호화 KEY', 'aes'), 'utf8');
postgres$# END; $$
postgres-# LANGUAGE PLPGSQL;
```

```
CREATE FUNCTION
postgres=# CREATE OR REPLACE FUNCTION TO_ENCRYPTS(data varchar)
postgres-# RETURNS varchar AS
postgres-# $$ BEGIN
postgres$#     RETURN encode(encrypt(convert_to(data, 'utf8'), '암호화 KEY', 'aes'), 'hex');
postgres$# END; $$
postgres-# LANGUAGE PLPGSQL;
CREATE FUNCTION
postgres=# CREATE OR REPLACE FUNCTION TO_DECRYPTS(data varchar)
postgres-# RETURNS varchar AS
postgres-# $$ BEGIN
postgres$#     RETURN convert_from(decrypt(decode(data, 'hex'), '복호화 KEY', 'aes'), 'utf8');
postgres$# END; $$
postgres-# LANGUAGE PLPGSQL;
CREATE FUNCTION
postgres=# CREATE OR REPLACE FUNCTION TO_ENCRYPTS(data text)
postgres-# RETURNS text AS
postgres-# $$ BEGIN
postgres$#     RETURN encode(encrypt(convert_to(data, 'utf8'), '암호화 KEY', 'aes'), 'hex');
postgres$# END; $$
postgres-# LANGUAGE PLPGSQL;
CREATE FUNCTION
postgres=# CREATE OR REPLACE FUNCTION TO_DECRYPTS(data text)
postgres-# RETURNS text AS
postgres-# $$ BEGIN
postgres$#     RETURN convert_from(decrypt(decode(data, 'hex'), '복호화 KEY', 'aes'), 'utf8');
postgres$# END; $$
postgres-# LANGUAGE PLPGSQL;
CREATE FUNCTION
postgres=#
```

1.5 암복호화 함수 생성 확인

```
postgres=# \df
                                        List of functions
 Schema |        Name         | Result data type |       Argument data types        |  Type
--------+---------------------+------------------+----------------------------------+------
-
 public | armor               | text             | bytea                            | normal
 public | armor               | text             | bytea, text[], text[]            | normal
 public | crypt               | text             | text, text                       | normal
 public | dearmor             | bytea            | text                             | normal
 public | decrypt             | bytea            | bytea, bytea, text               | normal
 public | decrypt_iv          | bytea            | bytea, bytea, bytea, text        | normal
 public | digest              | bytea            | bytea, text                      | normal
 public | digest              | bytea            | text, text                       | normal
 public | encrypt             | bytea            | bytea, bytea, text               | normal
 public | encrypt_iv          | bytea            | bytea, bytea, bytea, text        | normal
 public | gen_random_bytes    | bytea            | integer                          | normal
 public | gen_random_uuid     | uuid             |                                  | normal
```

```
 public | gen_salt              | text             | text                                | normal
 public | gen_salt              | text             | text, integer                       | normal
 public | hmac                  | bytea            | bytea, bytea, text                  | normal
 public | hmac                  | bytea            | text, text, text                    | normal
 public | pgp_armor_headers     | SETOF record     | text, OUT key text, OUT value text | normal
 public | pgp_key_id            | text             | bytea                               | normal
 public | pgp_pub_decrypt       | text             | bytea, bytea                        | normal
 public | pgp_pub_decrypt       | text             | bytea, bytea, text                  | normal
 public | pgp_pub_decrypt       | text             | bytea, bytea, text, text            | normal
 public | pgp_pub_decrypt_bytea | bytea            | bytea, bytea                        | normal
 public | pgp_pub_decrypt_bytea | bytea            | bytea, bytea, text                  | normal
 public | pgp_pub_decrypt_bytea | bytea            | bytea, bytea, text, text            | normal
 public | pgp_pub_encrypt       | bytea            | text, bytea                         | normal
 public | pgp_pub_encrypt       | bytea            | text, bytea, text                   | normal
 public | pgp_pub_encrypt_bytea | bytea            | bytea, bytea                        | normal
 public | pgp_pub_encrypt_bytea | bytea            | bytea, bytea, text                  | normal
 public | pgp_sym_decrypt       | text             | bytea, text                         | normal
 public | pgp_sym_decrypt       | text             | bytea, text, text                   | normal
 public | pgp_sym_decrypt_bytea | bytea            | bytea, text                         | normal
 public | pgp_sym_decrypt_bytea | bytea            | bytea, text, text                   | normal
 public | pgp_sym_encrypt       | bytea            | text, text                          | normal
 public | pgp_sym_encrypt       | bytea            | text, text, text                    | normal
 public | pgp_sym_encrypt_bytea | bytea            | bytea, text                         | normal
 public | pgp_sym_encrypt_bytea | bytea            | bytea, text, text                   | normal
 public | to_decrypts           | character        | data character                      | normal
 public | to_decrypts           | character varying | data character varying              | normal
 public | to_decrypts           | text             | data text                           | normal
 public | to_encrypts           | character        | data character                      | normal
 public | to_encrypts           | character varying | data character varying              | normal
 public | to_encrypts           | text             | data text                           | normal
(42 rows)
postgres=#
```

1.6 암복호화 함수 테스트

1) 암호화 함수(TO_ENCRYPTS)

```
postgres=# SELECT TO_ENCRYPTS('qwer1234');
            to_encrypts
----------------------------------
 2b9a5a43f720621f1c791f8882a75195
(1 row)
postgres=#
```

2) 복호화 함수(TO_DECRYPTS)

```
postgres=# SELECT TO_DECRYPTS('2b9a5a43f720621f1c791f8882a75195');
 to_decrypts
-------------
 qwer1234
```

```
(1 row)
postgres=#
```

2. About BaroCRYPT

Version 1.0 - Official Release - 2016.12.1

http://www.nurit.co.kr

상호: 주식회사 누리아이티
등록번호: 258-87-00901
대표이사: 이종일
대표전화: 02-2665-0119(기술지원/영업문의)
이메일: mc529@nurit.co.kr
주소: 서울시 강서구 마곡중앙2로 15, 913호(마곡동, 마곡테크노타워2)